DETECTION OF LOAD BALANCED LINKS IN INTERNET PROTOCOL NETWORKS

CROSS REFERENCE TO RELATED APPLICATIONS

Cross-reference is made to U.S. Patent Application Serial Nos. 10/127,888, filed April 22, 2002, entitled "Topology Discovery by Partitioning Multiple Discovery Techniques", to Goringe, et al., and 10/127,967, filed April 22, 2002, entitled "Using Link State Information to Discover IP Network Topology", to Goringe, et al., each of which contains subject matter related to the subject matter of the present application and is incorporated herein by this reference.

FIELD OF THE INVENTION

The present invention relates generally to network topology and specifically to ascertaining network topology in load balanced networks.

BACKGROUND OF THE INVENTION

The topology of a distributed processing network, such as an Internet Protocol network, is information with many potential uses in troubleshooting, administration, planning, and other tasks. "Topology" refers generally to the patterns of connection between the various machines in a network, and "topology information" refers to the body of information associated therewith. Complete and accurate network topology information can predict the path of an individual packet through the network and hence identify the set of network devices, such as routers and their associated interfaces, involved. Incorrect or

inaccurate information can lead to poor planning and/or administrative decisions, an inability to accurately measure network performance, and/or consumption of excessive resources in troubleshooting problems, particularly in IP telephony where jitter and packet round trip time can be crucial considerations.

Current solutions to the problem of topology discovery can be separated into two broad categories, namely Simple Network Management Protocol or SNMP-based and traceroute-based. The majority of current commercial products, such as Avaya ExpertNet™ and Hewlett Packard Open View™, use an SNMP-discovery mechanism to construct topology information. In this approach, topology information is obtained from the Management Information Base or MIB of one or more routers in the network segment or subnetwork of interest. The topology information in the MIB can, however, be incomplete and/or inaccurate. Neither of the two standardized routing tables available through SNMP, namely IpRouteTable (RFC1213) and IpCidrRouteTable (RFC 2096), are capable of containing the multiple routes, or redundant links, having the same metric or cost to a selected destination. Traceroute-based techniques are available on a number of major operating systems, such as Windows™, Unix, and Linux. Traceroute sends a series of Internet Control Message Protocol or ICMP packets, each with an increasing Time-To-Live or TTL, to a particular destination device. It then uses error messages that are returned from each router on-route when the TTL expires to construct the path to that destination. Although the traceroute technique can have an advantage over SNMP-based techniques, namely that traceroute takes into account the actual routing decisions that are made on packets traveling across the network, traceroute can return an incorrect path that is made up

of some combination of two or more physically separate paths, particularly when load balancing is in effect.

Both techniques are generally unable to detect the existence of load balancing, let alone the type of load balancing, in effect along a route. With reference to Fig. 1, there are two redundant links or paths depicted, namely the first path from router 100 to router 104 to router 112 and the second path from router 100 to router 108 to router 112. The redundant links permit traffic to be distributed across the multiple routes to use network resources more efficiently. In per-packet load balancing, each outgoing packet is queued to a router interface in a round-robin fashion. In other words, a first packet is sent along the first path, a second (next) packet along the second path, a third (next) packet along the first path, and so on. This type of load balancing can cause voice telephony packets to arrive out of sequence at the destination, which appears to the destination as jitter. In per-destination load balancing, each router interface caches the destination address of each packet such that the next packet addressed to the same destination will be sent down the same interface. In per-source/destination load balancing, better usage of per-destination of load balancing is obtained by caching both the destination and the source address at each router interface. This ensures that the next packet with the same to/from address pair is sent down the same interface. Load balancing can not only be detrimental to IP telephony but also cause difficulties in measuring link characteristics as it can become difficult determining exactly to which link the measured characteristics correspond.

SUMMARY OF THE INVENTION

These and other needs are addressed by the various embodiments and configurations of the present invention. The present invention is directed generally to a system and method for detecting load balancing in a distributed processing network.

In one embodiment, a method for detecting load balancing in a distributed processing network is provided that includes the steps of:

(a) providing a baseline topology;

(b) selecting, from the baseline topology, first and second addresses associated with first and second routers, respectively, such that the first router has an associated first hop count relative to a selected node and the second router an associated second hop count relative to the selected node, with the first hop count being less than the second hop count;

(c) transmitting one or more (typically at least two) test packets, with each one test packet having a time to live equal to or greater than the second hop count;

(d) receiving one or more responses (*e.g.*, TTL-expired error messages) associated with the test packets; and

(e) determining, based on the responses, whether load balancing is in effect at the first router. These steps are performed iteratively preferably on subnetwork-by-subnetwork and router-by-router bases.

The step of selecting the first and second routers typically includes the additional steps of:

(i) selecting a (first) subnetwork;

(ii) identifying a first set of unique addresses within the selected (first) subnetwork; and

(iii) creating a second set of unique addresses.

The second set of addresses is the union of the first set and a third set of router interface addresses associated with routers between the selected node and the selected subnetwork. The first and second addresses are included within the third set.

In configuring the test packets, the time to live is preferably equal to the second hop count and the second hop count preferably exceeds the first hop count by one hop. The test packets are typically transmitted from a common source node while the destination address in the packet headers is held constant for per-packet load balancing detection and varied for per-destination and per-source/destination load balancing detection. Load balancing is in effect when at least two different routers respond to the test packets.

In a typical application, the test packets for detecting per-packet load balancing are sent first, and the test packets for detecting per-destination or per-source/destination load balancing thereafter. The detection of per-packet load balancing before per-destination and per-source/destination load balancing can be important as otherwise it would be difficult to know what type of load balancing is detected by the first set of test packets.

The method can have a number of advantages over conventional topology discovery algorithms. For example, the present invention can not only generate an accurate and complete topology of a desired network segment or subnetwork but can also identify the existence of load balancing and determine the type of load balancing in existence. This knowledge can facilitate troubleshooting, administration, planning and other network related tasks. In particular in Voice Over IP or IP telephony, this knowledge can lead to substantial time and cost savings in post-installation in IP telephony troubleshooting.

These and other advantages will be apparent from the disclosure of the invention(s) contained herein.

The above-described embodiments and configurations are neither complete nor exhaustive. As will be appreciated, other embodiments of the invention are possible utilizing, alone or in combination, one or more of the features set forth above or described in detail below.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a block diagram of redundant links according to the prior art;

Fig. 2 is a block diagram of a hardware implementation of an embodiment of the present invention;

Figs. 3A-C collectively are a flowchart of an operational embodiment of the topology discovery agent;

Fig. 4 depicts a simplified IpRouteTable according to a version of SNMP;

Fig. 5 depicts intermediate data structures for identifying per-packet load and per-source/destination load balancing; and

Fig. 6 depicts intermediate data structures for identifying per-destination load balancing.

DETAILED DESCRIPTION

Before discussing the configuration and operation of the present invention, it is important to understand certain features of many routing protocols. A routercan be identified by a unique router ID in the case of certain protocols, and associated with a unique area ID.

A router itself typically (but not always) has no IP address. Rather, the interfaces associated with the router generally have IP addresses. An interface is a logical device belonging to a host such as a router than can be the attachment point of a link. Typically, an interface will have zero or one IP address and belong to a network. The interface will normally have an interface number and a network mask. A link contains two or more bindings of a source interface and a metric or cost. It is templated by the metric representation which is specific to the routing protocol and represents the cost for a packet to leave an interface. A link is typically associated with a cost metric and a routing protocol identifier. A network object represents a data network or subnetwork. It has an address and a mask and represents an address space in which a set of hosts is contained. A network object may derive its address and/or its mask from its member interfaces.

The Network Topology Discovery System

With this in mind, Fig. 2 illustrates an exemplary network topology discovery system 200 according to an embodiment of the present invention. The system 200 is configured to be connected to an access point of a computer network, such as to stub network, to send communications to and receive communications from hosts, typically routers. The system 200 is a software-controlled machine comprising a memory 204 and a processor 208. The memory can be any suitable type of information recording medium, such as magnetic, optical, and magnetoptical, configured as internal and/or external and/or primary and/or secondary storage. The processor can be any suitable microprocessor configured to run any suitable operating system, including MS-DOS, UNIX, MVS, OS/2, VM/SP, and WINDOWS™.

The memory 204 comprises a topology discovery agent 212 configured to determine not only network topology generally but also detect the existence of load balancing and the type of load balancing, a baseline topology 216 containing topology information to be used as the starting point in topology discovery and a discovered topology 220 containing topology information in the baseline topology and discovered during topology discovery.

The topology discovery agent 212 preferably, given the baseline topology, can construct a path to each of the network edges (or edge subnets) and represents the paths as a tree structure. Such trees are conceptually similar to the spanning tree structure for ethernet bridges or to IP multicast group trees. The output is a modified form of the output illustrated in Fig. 22 of copending U.S. Patent Application Serial No. 10/127,888, *supra*. The agent 212 effects this result by using multiple invocations of traceroute (or traceroute-like) algorithm, first to detect the presence of per-packet load balancing and second to detect other forms of load balancing. As will be appreciated, the use of more invocations of traceroute per router/router interface, the greater the probability of detecting load balancing on that router/router interface. For example, if load balancing is to be detected on router R_1 224 that is one hop away from the system 200 (the selected node), a test packet having a TTL of 2 (the number of hops to the subject router R_1 plus one more hop) is sent to a selected destination such that the packet will pass through the selected router R_1. Based on the TTL-expired response packets generated by router R_2 228 and R_3 232, the identity or address(es) associated with the (next) downstream router R_2/R_3 can be determined. After multiple similarly configured test packets are sent, per-packet load balancing at router R_1 can be identified by the presence of the different respondent routers R_2 and R_3. In a later invocation for the

selected router, packets having the same TTL but different destination addresses are sent to identify the presence or absence of per-destination and per-source/destination load balancing.

The baseline topology 216 is topology information collected and configured in any suitable manner. In one configuration, the baseline topology is obtained by accessing network device routing tables using SNMP IpRouteTable and/or IpCidrRouteTable entries. In this configuration, the topology information is possibly incomplete. Fig. 4 depicts a simplified baseline topology information obtained by these techniques. As can be seen from Fig. 4, the topology 216 comprises destination address 400 correlated with next hop address 404. In other words, the topology 216 provides the next hop address for each destination address on a received packet. As will be appreciated, there are some routes which are not contained in the topology information. In the baseline topology 216, there is such a table for each identified router (which is identified by a corresponding router identifier and/or one or more associated interface addresses). In another configuration, the baseline topology is obtained using standard traceroute techniques. As noted above, traceroute techniques alone can provide an incorrect topology, particularly where per-packet load balancing is in effect. In other configurations, the baseline topology is obtained using other techniques, such as described in copending U.S. Applications entitled "Using Link State Information to Discover IP Network Topology" and "Topology Discovery by Partitioning Multiple Discovery Techniques", identified above.

The discovered topology 220 is preferably in the form of a network tree that describes the set of paths accessible from the system 200. By way of illustration, Fig. 22 of Serial No. 10/127,888 is presented in the form of a network tree with nodes and interconnecting links and a subnetwork clouds illustrated. The system 200 will normally be the root or trunk of

the tree, the links the branches, and the edge hosts (or hosts at the network edges) the leaves of the tree. Nodes of the tree will be routers, or "clouds", namely sets of routers and paths for which per-packet load balancing was detected and for which no deterministic path could therefore by output. As noted, the agent 212 produces the tree by traversing the baseline topology starting from the system 200 and adding to the output tree as new routes are discovered. As will be appreciated, the discovered topology 220 can be rendered in any other desirable form, such as via a database or STL, a markup language such as Extensible Markup Language or XML, a proprietary file format, and the like.

The system 200 can include other modules (which are not shown). For example, the system 200 can include a metric measuring agent configured to measure delays, such as jitter and packet loss rate, experienced by packets traveling to and from each of the identified edge hosts in the network. When load balancing is in effect or otherwise present, the destination address of the directly connected upstream router is typically not varied when attempting to measure delays to/from intermediate routers. In other words, an intermediate router may not be pinged directly, as the path taken to that router may not be the expected one. A TTL-based method, similar to that used by traceroute techniques, is typically used to ping the intermediate routers for metric measurement. This approach will elicit an ICMP TTL-expired message when the ping packet reaches the intermediate router. As will be appreciated, the Uniform Datagram Protocol or UDP can also be used instead of or in addition to ICMP not only for metric measurement but also for load balancing detection. UDP packets are treated similarly to voice telephony or VoIP packets by most routers. By picking a UDP port which is typically not open on the edge host, a response can be elicited from the edge host, in the form of a port-unreachable ICMP error message.

Operation of the Topology Discovery Agent

Referring to Figs. 3A-C, the operation of the topology discovery agent 212 will now be discussed.

In step 300, the agent 212 reads the baseline topology 216 file, and generates a set S of subnet work addresses in the network topology. This can be effected based on the baseline topology. Typically, the baseline topology file includes a list of routers and, for each listed router, a table similar to the table of Fig. 4.

In step 302, a next subnet S_i in the set S of subnets is selected, and in step 304 a (first) set E of device addresses inside S_i are generated. The device addresses in set E can be determined based on the baseline topology and/or other topology discovery techniques. Additionally, the addresses can be generated by known techniques based on the selected subnet address as discussed in detail below. At minimum, the set E will include the interface address of the router upstream from the selected subnet. With reference to Fig. 2, the set E will include at minimum the interface of router R_4 236.

In step 306, a (second) set D of addresses is created. The set D is the union of the device addresses in set E and the router interface addresses between the testing node or system 200 and the selected subnet S_i (or a third set of unique addresses). With reference to Fig. 2, when the subnet 240 is S_i, set D includes the addresses of at least one associated interface 256a-j for each of routers R_1, R_2, R_3, and R_4. As will be appreciated, step 306 may be omitted. If a generated list of IP addresses is used (discussed infra), it is not necessary to construct the set D; instead, the addresses within set E are used.

In step 308, a next router interface address from set D is selected. The router interface address selected is preferably associated with the router immediately downstream from system 200.

In decision diamond 310, the router interface address is pinged by known techniques (*e.g.,* by SNMP or ICMP) to determine if the interface address is valid, *i.e.,* contactable. When the address is invalid, the agent 212 returns to step 302 above. To enable router utilization monitoring and baseline topology determination, SNMP and ping contactability are preferred for each router in the tree. If a router were not SNMP and ping contactable, the tree may be pruned at that point. SNMP contactability is normally not required for load balanced link detection alone. When the address is valid, the agent 212 proceeds to step 312.

In step 312, the agent 212 removes the selected and validated interface address and any associated interface address (for the selected router) from set D. Associated interface addresses exist where a given router has more than one contactable interface identified in the baseline topology.

In step 314, the agent 212 initializes and sets the following parameters:

(a) "h", or the hop count, is set equal to the hop count from the system 200 (or selected node) to the selected router interface address. For example, in Fig. 2 if the selected router interface address is an interface 256a of router R_1 224 the hop count "h" is set to 1 (as the router R_1 is one hop from the system 200 while routers R_2 228 and R_3 232 are each two hops from system 200, and so on).

(b) "R_u", or an address associated with the immediately upstream router from the selected router, is set to an interface address of the immediately upstream router.

In decision diamond 316, it is determined if there is an address to which R_u can be set. For example, if the initially selected router is router R_1, there is no router upstream of R_1 (as R_1 is the immediately downstream router from the system 200). In contrast if the selected router is router R_2 or $R_{3,}$ the upstream router is R_1. In the event that there is no address corresponding to R_u, the agent 212 returns to step 308, selects the next downstream router, which in the configuration of Fig. 2 is either router R_2 or R_3, and repeats the above steps. In the event that there is an address corresponding to R_u, the agent 212 proceeds to step 320.

In step 320, the agent 212 tests for per-packet load balancing on R_u by pinging the selected router a selected number ("Np") times with the Time To Live or TTL set to "h". Thus, for either router R_2 or R_3 as R_u "h" is set to two. A higher probability that load balancing has been detected is associated with a higher value of Np. Typically, Np is set to ten, or ten test packets are sent, to provide a greater than 90% probability that load balancing will be detected on the selected router. The destination for the packet can be any destination downstream of the selected router as well as the address of the interface of the selected router itself. Referring to Fig. 2, if router R_2 or R_3, is selected the packet destination can be an interface address associated with the selected one of router R_2 228 or R_3 232 or router $R_4$236, the address x.x.x.0 of subnet 240, or any of the addresses of S_1 (244), S_2 (248), . . . S_n (252). To detect per-packet load balancing, only one destination address is typically employed and that destination address is preferably an edge subnet address.

In decision diamond 324, it is determined whether a response is received to any of the test packets from the selected router. If not, the baseline topology file 216 is deemed to be incorrect and the agent 212 proceeds to step 396 and terminates operation. Although it

seems excessive to quit upon discovering inaccurate topology particularly if the router in question is attached to a relatively unimportant edge subnet, it is left to the user to sort out the problem as the network topology is probably fairly unstable or has changed since the baseline topology was acquired. If a response is received only from a router other than the selected router, the topology is nonetheless deemed to be incorrect. If a response is received from the selected router, the agent 212, proceeds to decision diamond 328.

In decision diamond 328, it is determined whether a response is received from a router other than the selected router. During transmission of the test packets a table similar to that of Fig. 5 is maintained. As can be seen from the figure (which depicts a per-packet load balancing test being performed on R_1 as the upstream router) , the table has a column 500 for respondent router and a column 504 for the number of responses or hits. As will be appreciated, the TTL for each of the test packets in the table is maintained constant while the destination is varied. When there is only one router sending responses to the test packets, per-packet load balancing is not deemed to be in effect. When there is more than one router sending responses to the test packets (which is the case in Fig. 5), per-packet load balancing is deemed to be in effect. When per-packet load balancing is in effect, the agent 212 in step 332 instantiates a "cloud" between R_u and the selected subnet and returns to step 302. A "cloud" is instantiated as further load balancing detection downstream of the selected router can provide an erroneous topology. When per-packet load balancing is not in effect, the agent 212 proceeds to decision diamond 336.

In decision diamond 336, the agent 212 determines whether the size or membership of set D is equal to (or less than) one. If only one member remains in set D, the agent 212 is typically unable to test for per-destination and per-source/destination load balancing; that

is, the agent is unable to test for per-destination and per-source/destination load balancing when set D contains all known or predictable destination addresses downstream of the selected router. In that event, the agent in step 340 instantiates a link between R_u and the selected router, with a warning indicating that per-destination and per-source/destination load balancing could not be tested for the associated link, and proceeds to step 374 discussed below. Whenever a link is instantiated, the agent 212 determines the input and output interfaces for each router. These are recorded in the tree for the purpose of router interface monitoring. If more than one member remains in set D, the agent 212 proceeds to step 344.

In step 344, the agent 212 tests for per-destination and per-source/destination load balancing on the upstream router R_u. This is typically effected by pinging all addresses in set D with the TTL equal to "h". Because the source address of the system 200 is generally constant among the test packets, per-destination and per-source/destination load balancing are effectively identical for purposes of detection and corrective action. In one configuration, other destination addresses are generated and used for test packets. Such addresses can be generated by selecting destination addresses off of the subnet address of subnet S_L. For example, as shown in Fig. 2, if the address x.x.x.0 for subnet 240 is known the addresses x.x.x.1 for destination S_1, x.x.x.2 for destination S_2, and x.x.x.n for destination S_n can be determined. Even though the generated addresses may not be valid, they may still be used as the test packet, due to the TTL, will not reach the address. Alternatively or additionally, destination addresses can be obtained from the router table of last router before the destination, such as router R_4 in Fig. 2. As will be appreciated, the more destination addresses employed in test packet headers during load balancing detection means a higher probability of detecting per-destination and per-source/destination load balancing.

During transmission of the test packets a table similar to that of Fig. 6 is maintained. As can be seen from the figure (which depicts a per-destination and per-source/destination load balancing test being performed on R_1 as the upstream router), the table has a column 600 for destination, a column 604 for respondent router, and a column 608 for number of responses or hits. When there is only one router sending responses to the test packets, per-destination and per-source/destination load balancing is not deemed to be in effect. As can be seen from Fig. 6, either per-destination or per-source/destination load balancing is in effect for the upstream router R_1 that is the subject for the test referenced therein.

In decision diamond 348, it is determined whether a response is received to any of the test packets from the selected router. If not, the baseline topology file is deemed to be incorrect and the agent 212 proceeds to step 396 and terminates operation. If a response is received only from a router other than the selected router, the topology is nonetheless deemed to be incorrect. If a response is received from the selected router, the agent 212, proceeds to decision diamond 352.

In decision diamond 352, it is determined whether a response is received from a router other than the selected router. If all responses are received from the selected router, the agent 212 assumes no load balancing is in effect and, in step 356, instantiates a link between R_u and the selected router and proceeds to step 374 below. If a response is received from a router other than the selected router, a decision must be made as to which of the load-balanced links to follow downstream.

In step 360, the decision as to which link to follow is effected. Although any technique can be used to effect the selection, a preferred technique is to set the downstream router R_d to the interface address of the router generating the most responses to pings in the

load balancing test of step 344. In Fig. 6, R_d is set to an interface address associated with R_2. In the event of an equal number of responses for each potential router, the router may be selected arbitrarily.

In decision diamond 364, the agent 212 determines whether or not R_d is in the baseline topology 216. If not, the baseline topology is incorrect and the agent proceeds to step 396 and terminates operation. If so, the agent instantiates a link between R_u and R_d in step 366 with a suitable warning indicating that per-destination or per-source/destination load balancing is in effect.

In step`368, all (destination) addresses that failed to return a response from R_d are removed from the set D. In other words with reference to Fig. 6 and assuming that R_d is set to an interface address of R_2, the address x.x.x.2 would be removed from the set D.

In step 370, the agent 212 sets the next router interface address is set to R_d and checks for asymmetry in step 374. As will be appreciated, asymmetric paths can arise in some routing protocols, such as Open Shortest Path First or OSPF, when the metric for one direction of a link is not the same for the reverse direction. This results in packets sent from node A to node B being forwarded through a different set of routers than packets sent from node B to node A. Such paths are undesirable in general for real-time communications, as the delay/packet loss/jitter characteristics for the return path of a packet may be substantially different to that for the outbound path. Detection of asymmetric paths can be performed given access to each router's routing tables. If an outbound path from node A contains a segment between first and second routers, the routing table of the second router is examined to ensure that for packets destined to node A, the next hop is the first router. Accordingly,

the agent 212 accesses the baseline topology 216 and finds the entry for the address of agent 212 in the routing table of the selected router.

In decision diamond 378, the agent determines if the next hop address in the entry is the selected router. If not, a warning is added in step 380 to the discovered topology for the selected router indicating that the associated link may be asymmetrical. The agent 212 then proceeds to step 388. If so, the agent proceeds to decision diamond 382.

In decision diamond 382, the agent determines whether the incoming link R_u to the selected router has a warning for load balancing. If so, the agent 212 in step 384 adds a warning indicating that the associated link may be asymmetrical. It is assumed that, when there is load balancing in the downstream direction, it is likely that there will be load balancing in the upstream direction. This gives rise to the possibility of asymmetry, hence the extra warning. Thereafter or if the incoming link has no warning, the agent 212 proceeds to decision diamond 388.

In decision diamond 388, the agent 212 determines whether there is a next router interface address in set D. If so, the agent 212 returns to step 308 and sets the next router address to the entry in set D. If not, the agent proceeds to decision diamond 392.

In decision diamond 392, the agent 212 determines whether there is a next (edge) subnet that has not yet been the subject of load balancing testing. When a next untested subnet exists, the agent 212 returns to step 302 and sets the next subnet S to the untested subnet. When no next untested subnet exists, the agent proceeds to step 394.

In step 394, the agent 212 writes the discovered topology tree to the discovered topology 220 output file(s) and terminates operation in step 396.

A number of variations and modifications of the invention can be used. It would be possible to provide for some features of the invention without providing others.

For example in one alternative embodiment, only a subset of set D is pinged in step 344. Although this reduces network traffic, it can also reduce the chances of detecting subsequent downstream load balancing.

In another alternative embodiment, instead of quitting in decision diamond 364 when R_d is not found in the baseline topology, another downstream router that is included in the baseline topology could be picked for R_d.

In yet another alternative embodiment, step 360 could be changed so that instead of following a single downstream link all of the downstream links are followed. This modification would increase coverage of the network but also the complexity of the software.

In a further alternative embodiment, in step 332 instead of instantiating a cloud the topology is "pruned" at the selected router using per-packet load balancing and no tests are performed on components downstream of the "pruned" router. "Pruning" refers to removal of the branch(es) of the tree downstream from a selected point.

In another embodiment, any of the software modules discussed above can be implemented, in whole or part, as an application specific integrated circuit or any other type of logic circuit.

The present invention, in various embodiments, includes components, methods, processes, systems and/or apparatus substantially as depicted and described herein, including various embodiments, subcombinations, and subsets thereof. Those of skill in the art will understand how to make and use the present invention after understanding the present disclosure. The present invention, in various embodiments, includes providing devices and

processes in the absence of items not depicted and/or described herein or in various embodiments hereof, including in the absence of such items as may have been used in previous devices or processes, e.g., for improving performance, achieving ease and\or reducing cost of implementation.

The foregoing discussion of the invention has been presented for purposes of illustration and description. The foregoing is not intended to limit the invention to the form or forms disclosed herein. In the foregoing Detailed Description for example, various features of the invention are grouped together in one or more embodiments for the purpose of streamlining the disclosure. This method of disclosure is not to be interpreted as reflecting an intention that the claimed invention requires more features than are expressly recited in each claim. Rather, as the following claims reflect, inventive aspects lie in less than all features of a single foregoing disclosed embodiment. Thus, the following claims are hereby incorporated into this Detailed Description, with each claim standing on its own as a separate preferred embodiment of the invention.

Moreover though the description of the invention has included description of one or more embodiments and certain variations and modifications, other variations and modifications are within the scope of the invention, e.g., as may be within the skill and knowledge of those in the art, after understanding the present disclosure. It is intended to obtain rights which include alternative embodiments to the extent permitted, including alternate, interchangeable and/or equivalent structures, functions, ranges or steps to those claimed, whether or not such alternate, interchangeable and/or equivalent structures, functions, ranges or steps are disclosed herein, and without intending to publicly dedicate any patentable subject matter.